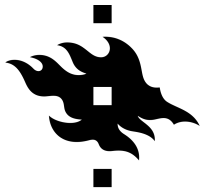

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Propulsion Power And Fuel Consumption Estimation Of Ships

Tomi Kallava

August 2019

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Mathematics and Statistics

Tekijä — Författare — Author
Tomi Kallava

Työn nimi — Arbetets titel — Title

Propulsion Power And Fuel Consumption Estimation Of Ships

Oppiaine — Läroämne — Subject
Mathematics

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	August 2019	41 p.

Tiivistelmä — Referat — Abstract

This thesis is done for Wärtsilä, which is a big global actor in marine and energy markets. This thesis aims to test the feasibility of machine learning models in estimating total power- and fuel consumptions of vessels' main engines and thus help in recognizing the effect of different factors on the energy consumption of vessels. This, on the other hand, helps to optimize routes and machinery concepts among other things. Another goal is to compress the engine sensor data utilizing wavelet transformation.

After the introduction to the topic in the second chapter, we introduce the data we are using in this study. These include vessel location data, engine sensor data and technical specifications of the vessels. In the third chapter, we go through the mathematical formulations of the used methods. Finally, we will perform the calculations with real data and analyze the results. We'll test the performance of compression methods applied to the time series data coming from sensors. After that, we'll test different regression methods for consumption estimations and see what gives the most accurate results.

Avainsanat — Nyckelord — Keywords
Haar wavelet, Linear regression, Random forest

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Matematiikan ja tilastotieteen laitos

Tekijä — Författare — Author
Tomi Kallava

Työn nimi — Arbetets titel — Title

Laivojen propulsiotehon käytön ja polttoainekulutuksen arviointi

Oppiaine — Läroämne — Subject		
Matematiikka		
Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu -tutkielma	Elokuu 2019	41 s.

Tiivistelmä — Referat — Abstract

Tämä työ on tehty Wärtsilälle, joka on iso kansainvälinen toimija merenkulku- ja energia-alalla. Työn tavoitteena on luoda koneoppimismalleja, joiden avulla voi arvioida eri muuttujien vaikutusta laivojen propulsiotehon käyttöön ja polttoainekulutukseen. Tämä puolestaan luo mahdollisuuksia optimoida esimerkiksi laivojen reittejä ja propulsiojärjestelmän kokoonpanoja. Osaongelma, joka tässä työssä myös pyritään ratkaisemaan, on laivan moottoreista saatavan anturidatan älykäs pakkaaminen. Tähän käytämme muun muassa ns. aallokemuunnosta.

Johdannon jälkeen toisessa luvussa perehdytään työssä käytettäviin tietoaineistoihin. Näihin kuuluvat laivojen sijaintitieto, moottoreiden anturidata sekä laivojen tekniset tiedot. Kolmannessa luvussa esitetään työssä käytettävien menetelmien matemaattiset muotoilut. Lopuksi käydään läpi kokeellinen osuus ja tulosten analysointi. Kokeilemme anturidatan pakkausmenetelmän toimivuutta sekä eri regressiomenetelmiä ja niiden antamien tulosten tarkkuuksia.

Avainsanat — Nyckelord — Keywords
Haar-aalloke, Lineaarinen regressio, Satunnaismetsä

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Acknowledgements

I want to thank people in Wärtsilä, especially Tal Katzav, Jarkko Pukkila and Inka Vilpola for the support in getting this thing started in the first place and Tuomas Sipilä for being the source of inspiration behind defining the topic for this thesis. I would also like to thank exactEarth and Clarkson PLC for the opportunity to use the data provided by them for the analysis.

I'd like to thank also Samuli Siltanen for supervising this work but also for previous efforts for enabling me to progress with my studies and learn a lot about computational mathematics.

Last but not least I would like to thank my family; Saana, Seela, Sofia and Milla and of course mom, dad and brothers Jussi and Markus.

Contents

Chapter 1

Introduction

Climate change is at hand and there's an urgent need for a set of technological solutions to reduce greenhouse gas emissions. The International Maritime Organization (IMO) has been estimating that carbon dioxide emissions from shipping covered 2,2% of all the emissions produced by human in 2012 and the portion has already increased since then and will still continue increasing if no action will be taken. Cruise ships alone produced 35 million tonnes of carbon dioxide in 2012.[1] Not to mention the infamous quote that has been spread in the news stating that the world's 15 biggest ships create more pollution than all the cars in the world. This is presumably related to nitrogen oxide and sulphur oxide emissions and mainly caused by usage of heavy fuel oil. We're not going to argue about the facts behind that statement in this thesis but it works as a reminder of how much we could potentially do in the maritime industry for enabling sustainable future.

Some of the opportunities in emission reduction of vessels are for example slow steaming (reducing sea speed), investing in alternative fuels like liquefied natural gas (LNG) and increasing energy efficiency with advanced ship design and intelligent voyage planning.

The main objective of this thesis is to facilitate data driven ship design and smart voyage planning by estimating propulsion powers and fuel consumptions of large vessels based on their movement. By data driven ship design we mean that when we have a rough idea of how the new build vessel would be operating and we know the measures of the vessel, we could estimate the distribution of needed propulsion powers and from there design the optimal machinery concept for the ship's propulsion system. In the smart voyage planning we could optimize for example the route and speeds to meet the expectations related to schedule, fuel consumption and so on.

The idea is to compare classic ways of calculating power/fuel consumptions to some machine learning based calculations and see if we can get more accurate results with these modern data driven methods. On the other hand, we could find some features that work as good predictors but which have not been taken into account in classical estimation

formulas.

There are some factors that certainly would effect to power consumption but where the data is not available for this study and/or the factors would be too vessel-specific to be utilized in larger scale models. These factors are at least weather conditions and frictional resistance between water and vessel's hull. In the latter case one big factor is vessel hull fouling. This is something that would probably be hard to measure and even harder to predict. Weather conditions data on the other hand is a low hanging fruit since there are several public APIs (application programming interfaces) that provide weather forecasts and historical weather data. It would be beneficial data for voyage planning. Nevertheless, the weather data is out of scope of this thesis due to, inter alia, additional costs it would produce.

As a ground truth and labelled training data we will use direct sensor data measurements from five large cruise ships equipped with Wärtsilä engines. Other data sources we have are Clarksons' data, which includes static information about vessels and AIS data, which is vessel location data provided by company called ExactEarth. Unlike the engine sensor data we have in use, AIS- and Clarksons-data cover all the largest vessels in the world so the estimations can be done based on those data sets.

Big part in creating any machine learning algorithm is collecting the data and cleaning, compressing, standardizing and grouping it in suitable ways. These steps also include some technicalities and mathematical formulations which we will focus on before the actual model fitting. But even before that we will go through the data sources we are using; where the data sets came from, who produced them and what were they meant for originally?

Chapter 2

Data Sources

2.1 Vessel location data

Automatic Identification System (AIS) is a tracking system for vessels. It was developed by the IMO (International Maritime Organization) technical committees. AIS transciever is mandatory for all vessels with weight over 300 tons. AIS was originally intended to allow ships to view marine traffic in their area and to be seen by other traffic and that way to avoid collisions between large vessels.

For this thesis, we are getting snapshots of AIS data from about 100000 largest vessels every ten minutes. Data contains some static information we don't care about because we have more reliable and comprehensive source for that kind of data. The dynamic parameters we care about or might care about are listed below:

- "timestamp": Important for combining data with other time series data
- "sog": Speed over ground
- "draught": Draught is the vertical distance between the waterline and the bottom of the vessel's hull so it tells how deep in the water the ship sails.[2]
- "latitude": geographic coordinate that specifies the north–south position of a point on the Earth's surface
- "longitude": geographic coordinate that specifies the east–west position of a point on the Earth's surface
- "destination": Name of the port/city where the vessel is heading
- "heading": Direction where the vessel's bow is heading. 0-359 degrees relative to north. Data is derived from gyro compass.

- "cog": Course Over Ground. Direction where the vessel is moving relative to north (degrees). This may differ from heading for example due to wind.
- "rot": Rate of turn (degrees per minute)

Although we should get data from every vessel every ten minutes, the reality is something else many times. Most of the vessels have at some point large gaps in the data flow probably mostly due to some data transmission issues. On the other hand, there may also be data quality issues related to parameter values. Reasons behind these issues are not covered here but it is good to acknowledge that there may be need for some sanity checks at some point when processing the data.

2.2 Static vessel data

Clarkson PLC, referred as Clarksons is a large shipping service provider. Part of the Clarksons group is Clarksons Research, which provides data and intelligence for global shipping. Clarksons provide data on over 135000 vessels.[5]

Parameters that are taken into consideration from the Clarksons' database are

- "imonumber": IMO-number (International Maritime Organization) is a unique identifier for ships which is used for mapping data sets together in this study.
- "vesseltype": Vessel type of the vessel can be defined in different granularity levels. For example Bulk Carriers, that are designed to carry unpackaged bulk cargo, can be specified to be Coal Carriers, Grain Carriers etc. but we will stick with higher level types like Bulk Carrier, Cargo Vessel, Cruise Ship and so on.
- "deadweight": Deadweight or deadweight tonnage is a measure of how much weight a ship can carry (tonnes) [9]
- "draft": This defines the draught that the ship is designed to operate with.
- "breadth": The maximum breadth over the extreme points between port side and starboard of the ship. [7]
- "loa": Length overall. Maximum length of a vessel's hull measured parallel to the waterline.
- "speedknots": Designed sea speed of the vessel.
- "mainpowerunitkwest": Nominal propulsion power of the vessel. That is the total power of main engines of the vessel.

2.3 Sensor data

There are tens, even hundreds of different values that are measured from each engine. For example, a lot of temperature measures are taken from different parts of the engine. All these are usefull for example for anomaly detection for predicting potential failures of the engine. Values relevant for this excercise are power and fuel consumptions of engines. Those values have to be aggregated to vessel level. In other words, measurements from each engine of one vessel has to be summed up.

In more detail, the values we collect from sensor signals are

- "power": Power in kilowats produced by engine
- "SFOC": Specific fuel oil consumption (g/kWh)
- "nbr_of_engines_running": How many engines are running at the moment

To calculate the absolute fuel oil consumption, we need to just multiply power with SFOC and then convert the unit of measure to anything we like. Suitable unit for fuel consumption is chosen to be tons/day, so

$$Fuel\_consumption = \frac{power * SFOC * 24}{1000000}(\frac{tons}{day}). \tag{2.1}$$

The reason we need to take into account also the information of the number of running engines is that we have to have valid measurements from all the engines that are running at that time in order to make reasonable estimations and this data provides an easy way to check that.

2.3.1 Main- and Auxiliary Engines

Main engines produce the propulsive power of vessels so they enable the movement of ships. Auxiliary engines on the other hand work as electricity generators. The higher the speed, the larger portion of power/fuel consumption is caused by main engines.[6]

Our aim is to estimate the propulsion power and the resulting fuel consumption. With the features used in this study, it is not feasible to estimate how much auxiliary engines are used. It is also not relevant because the energy efficiency optimization ideas considered here are related to vessel movement. Also when talking about data-driven machinery concept design, we are referring to propulsion systems. So we are taking into account only the main engines of vessels.

Chapter 3

Methods

Before diving into the experiments we will go through the mathematical formulations of each used method. We will go through the methods in chronological order and we will also at this stage already describe what each method is used for in the process.

3.1 Operating profile

One important aggregation from AIS-data that is related also to this work is operating profile. Vessel's operating profile is a distribution of modes of operation of a vessel. In this context we define it so that it is ship's speed and draught distributions in a selected time period. These are calculated from AIS-data. In practice we select evenly spaced intervals of speeds/draughts and count the number of points belonging to each interval. Then we divide the number with total amount of data points. The last bin of the histogram includes all the values above some selected maximum value. In speeds we have selected the maximum as 30 knots and maximum draught is selected to be 20 meters. This way we form speed/draught probability vectors for the vessel where every element of the vector indicates the probability of a certain speed-/draught interval. These are demonstrated in the pictures below. These two graphs are very revealing indicators of vessel's typical behaviour.

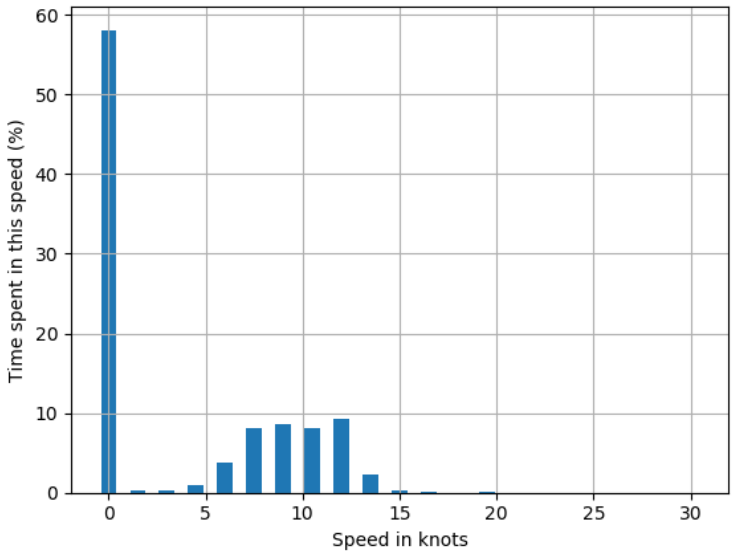

Figure 3.1: Speed profile of example vessel based on speeds of October 2018

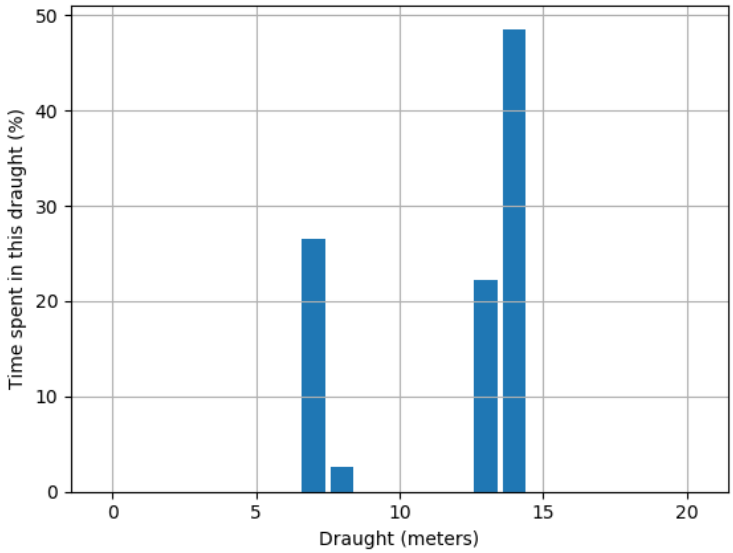

Figure 3.2: Draught profile of example vessel based on speeds of October 2018

The example vessel here is a bulk carrier so it's designed to transport bulk cargo. From the draught profile you can easily infere how much of the time the vessel has been loaded. When there's cargo, the draught is larger compared to when it's empty. In this case we could say that the vessel has been loaded about 70% of the time.

Many times more interesting than draught- and speed distributions themselves is that how those values occur together. For example in case of a bulk carrier you might want to know the speed distributions separately for each loading condition (loaded, ballast). To solve this issue, we collect the data points to matrix, where other axis is speed intervals and other axis is draught intervals. Now we let the width of each element in the matrix to be 2 meters (draught) and height to be 3 knots (speed). So we will end up in $10x10$ matrix. We can choose any other resolution as well. Every collected data point is put in the correct cell and the resulting value of the element will be the portion of data points placed in that element.

A heat map is created to better visualize the end result. Darker colours mean higher values. Those two main loading conditions (laden, ballast) are here also clearly visible. Laden is a condition where vessel is carrying cargo and ballast is a weight added to a vessel to increase draught and maintain the vessel in a safe condition of stability when there is no cargo on board. In other words we can say that ballast condition means empty vessel. We can assume that this vessel is carrying cargo when the draught is somewhere there above 13 meters.

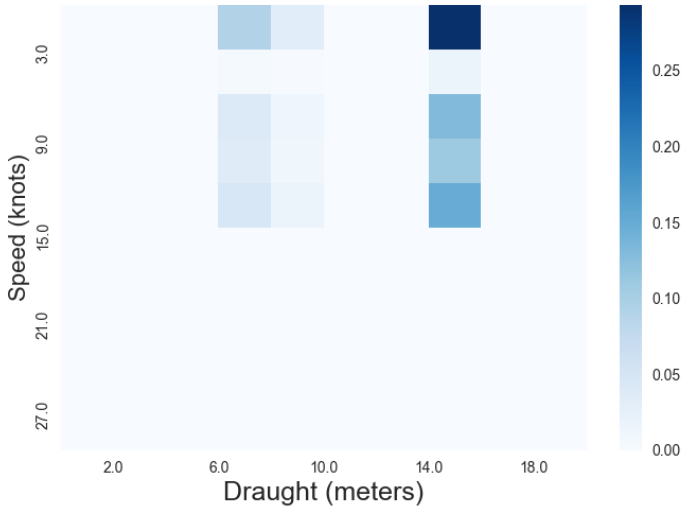

Figure 3.3: Speed-, draught matrix of example vessel based on data of October 2018

Why we care about operating profiles in this exercise? For example if we would like to estimate power demand or average daily fuel consumption for a new build vessel, we could estimate it if we roughly know the operating modes of a vessel. We could for example find so called sister vessels (some vessels that are assumabely operating in the same way that the new build would) and calculate their profiles. Instead of point-wise estimations we could do more general estimations by using profiles.

3.2 Haversine formula

When calculating vessel operating profiles there might be long gaps between data points and the speed information of AIS-data may have false values at times, instead of pointwise speed information we use average speeds between consecutive data points. Those can be calculated by dividing travelled distance by time difference between those points.

The distance between two points we have to infere from their latitude-longitude-coordinates. There we have to utilize spherical trigonometry and the so called haversine formula. That determines the distance between two points on a surface of a sphere.

Haversine formula

$$hav(\Theta) = hav(\varphi_1 - \varphi_2) + \cos(\varphi_1)\cos(\varphi_2)hav(\lambda_2 - \lambda_1), \tag{3.1}$$

where φ_1 and φ_2 are latitudes of points 1 and 2 and λ_2 and λ_1 are longitudes of points 1 and 2.

$$\Theta = \frac{d}{r}, \tag{3.2}$$

where d is the spherical distance between the two points and r is the radius of the sphere.[4] To solve the distance d, we have to apply inverse haversine

$$\begin{aligned} hav^{-1}(hav(\Theta)) &= \frac{d}{r} \\ &=> d = rhav^{-1}(hav(\Theta)) \\ &=> d = rhav^{-1}(hav(\varphi_1 - \varphi_2) + \cos(\varphi_1)\cos(\varphi_2)hav(\lambda_2 - \lambda_1)) \end{aligned} \tag{3.3}$$

where haversine function

$$hav(\theta) = sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}. \tag{3.4}$$

Now, the distance between two points on a surface of a sphere based on their latitudes and longitudes can be determined by

$$d = 2r \arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + cos(\varphi_1)cos(\varphi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right) \tag{3.5}$$

Earth is not a perfect sphere and for that reason haversine is not the most accurate way of calculating distances but for our purposes it is accurate enough. If we would need more accurate results we could use some iterative methods, like Vincenty's formulae that assume the figure of the earth to be oblate spheroid rather than a sphere. That would also add more complexity to measurements and is probably not worth the effort in this case.

3.3 Running Median

The time series data from sensors is noisy and it is too dense for our purpose so we have to remove the outliers and compress the signal somehow. Running median was discovered to be useful step for removing some of the erroneous measurements from signals.

Median is a value that divides the probability distribution so that randomly picked value from the data set is equally likely to be above or below the median value. In discrete case the median is $\frac{n+1}{2}th$ value of ordered list of values, where n is the number of values in the list. In case there's an even number of values in the set, the median is $\frac{\frac{n}{2}th+\frac{n+2}{2}th}{2}$.

Median value is more robust than average value because it's not skewed by extremely high or low values.

By running median we mean that the data is divided to chunks and the median value of each chunk is calculated. We can freely select the amount of data points in each chunk so that it fits the purpose best. Then all the chunks are replaced with the median.

3.4 Wavelet transformation

Wavelet transformation is a relatively modern tool in signal processing. It is used for example in compressing data or for extracting information from data. Wavelet transformation is often compared to perhaps more known method called Fourier transformation. The main advantage of wavelets is that they are generally localized in both time and frequency whereas the standard Fourier transform is only localized in frequency.

Wavelet is a wave-like oscillation that can be usually visualized with a short oscillatory wave (similar like for example heartbeat). The idea of wavelet transformation is that one can express the given function as a linear combination of wavelets (or wavelet coefficients) that together form an orthogonal basis of function space. These wavelets are all similarly shaped little bursts of signals but in different scales.

There are different kind of wavelets to choose from. The selection is usually made based on what mother wavelet represents the original signal best. Mother wavelet is a prototype function, in other words the basic building block of the wavelet decomposition.

Some mother wavelets are shown here:

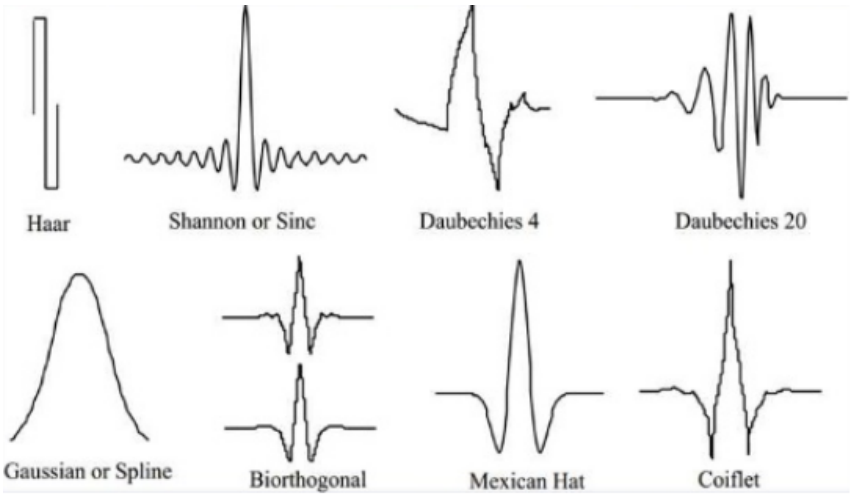

Figure 3.4: Some of the most common wavelets

3.4.1 Haar Wavelet

Haar wavelet is the simplest form of wavelets. It is ideal for compressing signal that is piecewise constant in nature. Power consumption of engine is assumed to be approximately piecewise constant signal (at least in operating modes that are relevant for this study). Haar wavelet appeared to be a good method for making a square-shaped approximation of the signal. The idea is to reduce the size of the data significantly without losing much information.

Mother wavelet of Haar is described as

$$\psi(t) = \begin{cases} 1, \ 0 \leq t < \frac{1}{2} \\ -1, \ \frac{1}{2} \leq t < 1 \\ 0, \ otherwise \end{cases} \tag{3.6}$$

and scaling function as

$$\varphi(t) = \begin{cases} 1, \ 0 \leq t < 1 \\ 0, \ otherwise \end{cases} \tag{3.7}$$

Scaling function's purpose is to ensure that whole spectrum is covered because wavelet transformation halves the bandwdith every time it's applied.

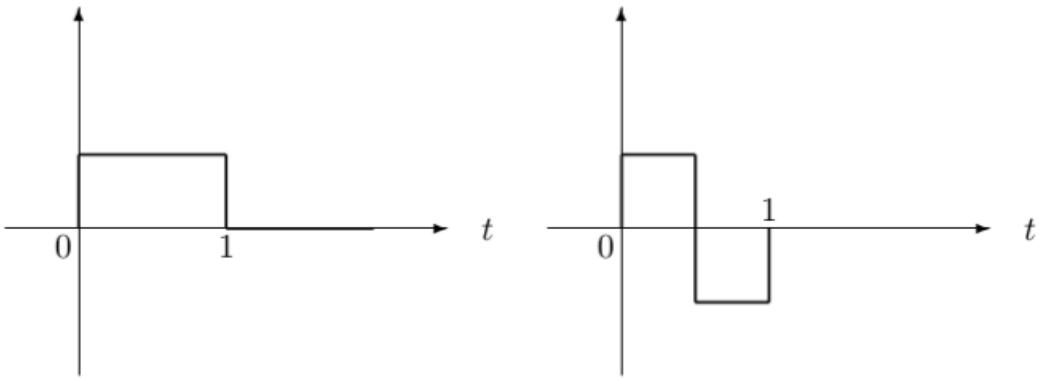

Figure 3.5: Haar scaling function and mother wavelet

Haar functions

$$\psi_{n,k}(t) = 2^{\frac{n}{2}}\psi(2^n t - k),\ t \in \mathbb{R}, \tag{3.8}$$

where n is the non-negative integer and $k \in [0, 2^j - 1]$, form the orthonormal basis of space V_n. V_0 is called the reference space and $...V_{-1} \subset V_0 \subset V_1 \subset V_2...$, where each V_n is spanned by the corresponding Haar function. How many of these functions are added up together defines the resolution of the reconstructed signal. [10][11]

3.4.2 Adaptive wavelet filtering

For noisy signals it is useful to add extra denoising step to selected details after the wavelet decomposition. A popular method for this is the so called SURE shrinkage (Stein Unbiased Risk Estimation). It is similar to lasso criterion, which is presented later on. It also starts with the criterion

$$\min_{\theta} ||\boldsymbol{y} - \boldsymbol{W}\boldsymbol{\theta}||_2^2 + 2\lambda||\boldsymbol{\theta}||_1, \tag{3.9}$$

where $\boldsymbol{y}$ is the original signal, $\boldsymbol{W}$ is the orthonormal wavelet basis matrix, $\boldsymbol{\theta}$ is the wavelet coefficient matrix and *lambda* is the weight coefficient of the penalty term. Because $\boldsymbol{W}$ is orthonormal, this leads to the equation where j^{th} coefficient

$$\theta_j = sgn(y_j^*)(|y_j^* - \lambda|), \tag{3.10}$$

where y_j^* is the noisy coefficient of the j^{th} decomposition level. The standard choice for $\lambda = \sigma\sqrt{2\log N}$, where σ is (estimate of) the standard deviation of the noise and N is the length of the coefficient vector i.e. number of samples collected from the signal. [11][12]

3.5 Admiralty Coefficient

Admiralty coefficient is used in the preliminary estimations of the power required in a new ship design to attain the desired speed. After finding out the block coefficient, we can modify the equation to estimate the power demand based on operating speeds. This is the formula we are competing against with our machine learning models. So can we estimate power demand better than with this formula? Admiralty coefficient

$$C = \frac{D^{\frac{2}{3}}V^3}{P}, \tag{3.11}$$

where D is displacement in tons, V is speed in knots and P is power kilowatts. Displacement refers to the mass of the water that the ship's hull displaces. We don't have direct displacement information available but that can be estimated with draught value and typical block coefficient value of the vessel type at hand. Block coefficient tells about the shape of the hull. The block coefficient of a ship is the ratio of the underwater volume of ship to the volume of a rectangular block having the same overall length, breadth and depth. Displacement can be now defined as

$$D = length * breadth * draught * block\_coefficient. \tag{3.12}$$

[7]

3.6 Regression Analysis

We are using regression methods to estimate power- and fuel consumptions of vessels. Regression is a statistical process to estimate relationships among variables. The idea is to estimate some variable that is dependent on some known inputs. These inputs are called for example predictors, regressors or, in machine learning terms, features. Best predictors are those that are dependent on the output but independent on other predictors.[12]

3.6.1 Occam's Razor

"Everything should be made as simple as possible, but not simpler"
- Einstein

Occam's razor is a principle in problem solving that states that the simpler solutions tend to be the correct ones. In machine learning and statistics it applies in a way that there's a fine line between a model being too simple and too complex. If the model is too simple we are talking about underfitting and with too complex models we are talking

about overfitting. Underfitting models are too rough to make good enough estimates and overfitting models work well with training data but don't generalize well enough so that they would work with unseen data. For this reason we will start with simpler models and work our way to more complex ones to find the best one for our case.

3.6.2 Mean Absolute Percentage Error

Mean absolute percentage error is chosen to be the metric to measure and compare the performances of the models. For optimisation of individual models we use different methods and this shouldn't be confused with optimization methods. Mean absolute percentage error is commonly used because of its easy and intuitive interpretation. It is defined by the formula

$$\boldsymbol{M} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|, \tag{3.13}$$

where A_t is the actual value, F_t is the forecasted value and n is the number of evaluated data points.

3.6.3 Indicator Variables

Some of the feature variables used for creating the model may be categorical. That means that the values are not real numbers but categories. For example speed is continuous quantitative variable unlike vessel type, which is a categorical variable. For these to work together in the same model we have to first of all somehow transform categorical values in a suitable form.

There we use a method called one-hot encoding. This method is borrowed from digital electronics, where one-hot stands for a group of bits among which the legal combinations of values are only those with a single high (1) bit and all the others low (0) [8].

In practice, categorical features are re-organized so that we take all the different values of one categorical feature and make a separate feature out of all distinct values. Finally, each of the data points has all of these features with one of the values being 1 and rest being 0.

3.6.4 Feature standardisation

Due to the fact that different features may have very different kind of scales of values, we have to somehow normalize them so that none of the variables get any wrong kind of weightings and by default all the feature variables will be treated equally. The solution is to standardize all the feature values.

The idea of standardization is to rescale the data so that the mean of values of a feature is zero and standard deviation is one. So the new value would be

$$x_{new} = \frac{x_{old} - \mu}{\sigma}, \tag{3.14}$$

where μ is the mean and σ is the standard deviation of the data set.

3.6.5 Linear Regression

Linear regression is a method to study relationships of variables. Simple linear regression gives an equation that approximates a relationship between two variables that are dependent on each other. Higher dimensional models take vector as an input but response variable is still scalar. Input vector that contains explanatory variables should contain variables that are independent but every variable in the vector should be dependent on the target variable. The third version is multivariate linear regression where target variable is a vector rather than scalar. In this study our goal is to predict scalar values with given feature vectors.

In multiple linear regression given the data set

$$\{y_i, x_{i1}, x_{i2}, ..., x_{ip}\}_{i=1}^{n} \tag{3.15}$$

with n data points and p features, the model assumes linear dependency between target variable y_i and corresponding vector of regressors $\boldsymbol{x_i}$. Usually the model includes some error term ϵ modeling random noise in the data. Then the linear regression model takes the form

$$y_i = \beta_0 1 + \beta_1 x_{i1} + ... + \beta_p x_{ip} + \epsilon_i = \boldsymbol{x}_i^T \boldsymbol{\beta} + \epsilon_i \tag{3.16}$$

Adding all the equations together we get the general notation

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{3.17}$$

where

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \tag{3.18}$$

$$\boldsymbol{X} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}, \tag{3.19}$$

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \tag{3.20}$$

and

$$\boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix} \tag{3.21}$$

In context of machine learning the goal is to create a linear regression model that predicts the outcome of the target variable of unseen data given the values of corresponding feature variables. The model is created by optimizing the parameters with help of labelled training data.

The Residual Sum Of Squares

The optimal linear regression model is found by minimizing the sum of squared residuals which denotes the sum of squared differences between the predicted and true values. So the goal is to find parameters $\boldsymbol{\beta}$ that minimizes

$$\begin{aligned} RSS(\beta) &= \sum_{i=1}^{N} (y_i - f(x_i))^2 \\ &= \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_i j \beta_j)^2. \end{aligned} \tag{3.22}$$

This equation can be compiled in to form

$$RSS(\beta) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}). \tag{3.23}$$

This function can be minimized by differentiating with respect to $\boldsymbol{\beta}$ and setting it to zero:

$$\begin{aligned} \frac{\delta RSS}{\delta \boldsymbol{\beta}} &= -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) = 0 \\ &=> \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) = 0 \end{aligned} \tag{3.24}$$

Now we will obtain the unique solution

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{3.25}$$

Now we are able to make a prediction given the new input vector $\boldsymbol{X}$:

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}} \tag{3.26}$$

Polynomial Features

Linear regression model can be extended by constructing polynomial features from the coefficients. For example if we have simple linear model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{3.27}$$

but we want to fit second order polynomial instead, we can combine features so that the model will take this kind of form:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 x_1^2 + \beta_5 x_2^2 \tag{3.28}$$

What might be unintuitive is that this is still a linear model. We just increased the number of features by creating new combinations from the existing ones. So, in this case, instead of original model with two features $[x_1, x_2]$ we have now five features $[x_1, x_2, x_1 x_2, x_1'^2, x_2^2]$.

Lasso Regularization

Lasso (least absolute shrinkage and selection operator) is a regularization method to increase the prediction accuracy and interpretability of the model. Lasso simply performs a variable selection so that it will shrink certain parameters to zero so that they will not play any role in the final model. In that way it will handle the problem of correlated inputs or inputs that have only minimal effect to the output. It forces the sum of the absolute value of the regression coefficients to be less than a fixed value, which forces certain coefficients to be set to zero. When utilizing lasso, we will add a condition to residual sum of squares that needs to be minimized:

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_i j \beta_j)^2, \text{ where } \sum_{j=1}^{p} |\beta_j| \leq t, \tag{3.29}$$

where t is the specified parameter that determines the amount of regularization. Similarly as RSS equation, this can also be expressed more compactly as

$$\min_{\boldsymbol{\beta}}\{\frac{1}{N}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2\}, \text{ where } ||\boldsymbol{\beta}||_1 \leq t \tag{3.30}$$

Finally, the most useful way of expressing lasso regularization when doing optimization is the so called Lagrangian form

$$\min_{\boldsymbol{\beta}}\{\frac{1}{N}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1\}. \tag{3.31}$$

Constraint region in Lasso is defined by ℓ^1-norm. That means that Lasso restricts the coefficients to a square shape. Like seen in the below picture, it tends to drive small weights to zero because there is a high probability of the optimal point to hit the corner point of unit sphere.

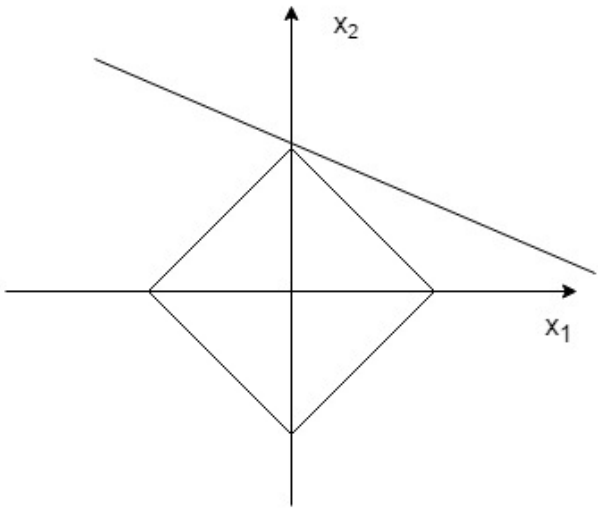

Figure 3.6: Unit sphere in L^1-space

3.6.6 Random Forest Regression

Random forest is one of the most effective learning algorithms out there. But still it is transparent and relatively easy to understand, which are very important factors when doing prescriptive analytics, where we need to understand how different factors influence to the end result.

Random forest is an ensemble method, which means that it aggregates results from ensemble of simpler estimators. These estimators in case of random forests are called decision trees. So, to explain how random forest works, we need to first enter into the world of decision trees.

Decision Trees

Decision tree is a flowchart like structure, where the data is broken down to decision nodes. The fundamental idea of decision tree is very simple. Here is one basic example of a decision tree that explains itself well:

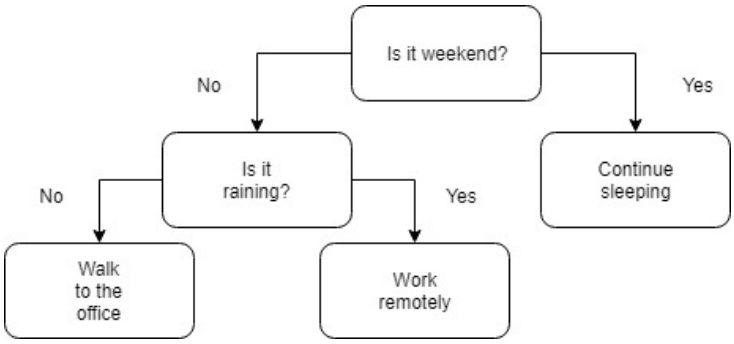

Figure 3.7: Simple decision tree

This was a very simple classification decision tree and an intuitive introduction to decision trees but what we are interested instead in this study is a regression model based on decision trees, where the features are mostly continuous numerical values. Here is a simplified example of regression tree where the feature parameters are standardized:

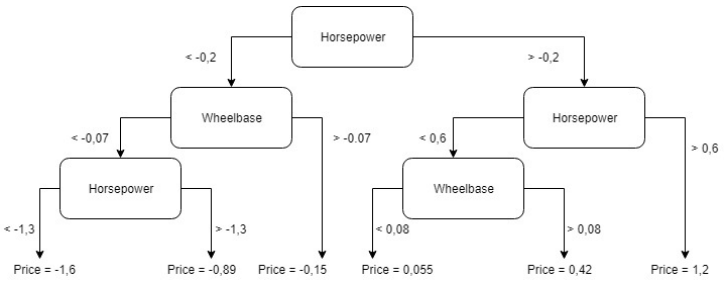

Figure 3.8: Standardized regression tree for price prediction of cars

If we put this into more formal form, where the outcomes instead of Price$= -1.5$, Price$= -0.98...$ would be R_1, R_2..., horsepower $= X_1$ and wheelbase $= X_2$, we'll get a regression model that predicts Y with constant c_m in region R_m:

$$\hat{f}(X) = \sum_{m=1}^{6} c_m I\{(X_1, X_2) \in R_m\} \tag{3.32}$$

Here the number of features and regions were still pre-defined according to the car price example. We can put this model in even more general format:

$$\hat{f}(X) = \sum_{m=1}^{M} c_m I\{x \in R_m\} \tag{3.33}$$

Like in linear regression, residual sum of squares is typically applied also in regression trees as a criterion for minimizing the error. so, if

$$\sum (y_i - f(x_i))^2, \tag{3.34}$$

we can see that optimal c_m is found by taking the average of y_i in region R_m:

$$\hat{c}_m = avg(y_i | x_i \in R_m) \tag{3.35}$$

Finding the optimal partitions of the tree with residual sum of squares is typically computationally too intensive. That's why usually greedy algorithm is applied.

Greedy Algorithm

Greedy algorithm is an algorithmic paradigm, where the idea is to make the locally optimal choices at each stage with the intent of finding the global optimum. [15]

The solution is usually not optimal but approximates the optimal solution well enough and most importantly, takes reasonable amount of time to execute.

Building the tree starts with splitting the whole dataset with splitting variable j and split point s to two half-planes

$$R_1(j,s) = \{X | X_j \leq s\} \text{ and } R_1(j,s) = \{X | X_j \leq s\}. \tag{3.36}$$

Then we will find j and s that solve

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \tag{3.37}$$

and the inner minimizations are solved by

$$\hat{c}_1 = avg(y_i | x_i \in R_1(j,s)) \text{ and } \hat{c}_2 = avg(y_i | x_i \in R_2(j,s)). \tag{3.38}$$

When the best split is found, then we will do the same process to two resulting regions, then to four resulting regions and so on until the tree reaches its pre-defined depth.

Bootstrap aggregation

Random forest fits multiple decision trees with various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but with bootstrap the samples are drawn with replacement. In other words bootstrap aggregation (also called bagging) is a special type of model averaging. More formally, given a standard training set $\boldsymbol{D}$ of size n, bagging generates m new training sets $\boldsymbol{D_i}$, each of size n', by sampling from D uniformly and with replacement.

3.7 Cross-Validation

Cross-validation is a technique for testing how a statistical model generalizes to an independent dataset. There are two objectives for cross-validation in this study; tuning the hyper parameters of individual models for creating the best possible model (grid search cross validation) and the other objective is to evaluate performances of the models.

Grid search cross validation is an exhaustive search over a set of hyper parameter combinations of a model for finding the combination that gives the best accuracy to the model. For example in random forest the parameters to optimize might be number of trees in the forest, the number of features to consider when looking for the best split, the minimum number of samples required to split an internal node and whether bootstrap samples are used when building trees or not. We pre-define some values for each of these parameters and then test all of the combinations and how they affect to the model accuracy.

K-fold cross validation is used for getting a reliable accuracy estimation for the model. If we randomly divide the data into training and test sets, train the model with training set and test the performance with the test set, we get the estimation of how the model performs. The problem might be the variance in prediction accuracy. This problem is highlighted if the test set is small or if many different models are tested out. To get stable performance metrics it is good to do some cross-validation so that the model is tested many times. In every testing round different subset of the data should be picked as a test set.

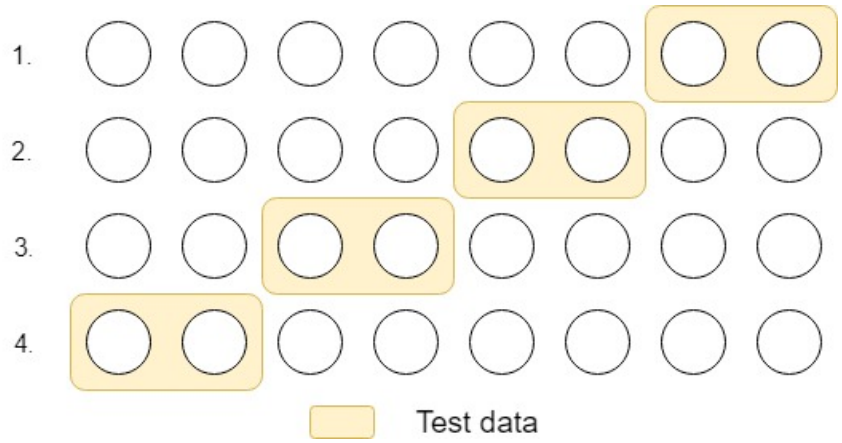

Figure 3.9: Four-fold cross validation. Training- and test sets are different on each round.

Chapter 4

Results

This chapter covers the experimental part of the thesis. We will start by cleaning and compressing the sensor data. Then we will combine data from different sources. Lastly we will do the estimation models and calculate the performance metrics for the models. We will do a general model over all vessels by randomly picking four out of five vessels and the data from those vessels will work as a training data for the models. Then we will test the models with the data from the remaining vessel.

We will test the models starting from the simplest one. The models we will test are linear regression, lasso, polynomial regression (linear regression with polynomial features) and random forest.

4.1 Coverage and limitations

We have collected data from five large cruise vessels from time period 31.5.2018-10.8.2018. This leads to the limitations of this experiment; we can only apply this model for cruise vessels since other types of vessels may work very differently. For example in cruise ships the draught is more or less constant while in cargo ships and bulkers it varies depending if the vessel is loaded or not. Also the block coefficients vary by vessel type. So this study works as a proof of concept but the solution can be easily extended to more vessels and vessel types when the data is available.

4.2 Sensor data cleaning and compression

The granularity of the sensor data gathered from the source system is one data point every 30 seconds. We would like to compress that data and also remove the outlier peaks because we are more interested in the general operating modes of the engine.

Signal had to be smoothed with running median before applying the main compressing method so that it won't be corrupted by thin peaks which we consider noise. Suitable kernel size (median filter window) was found by experimenting with different values and it was 7. Running median filtered signal together with the raw signal is shown below.

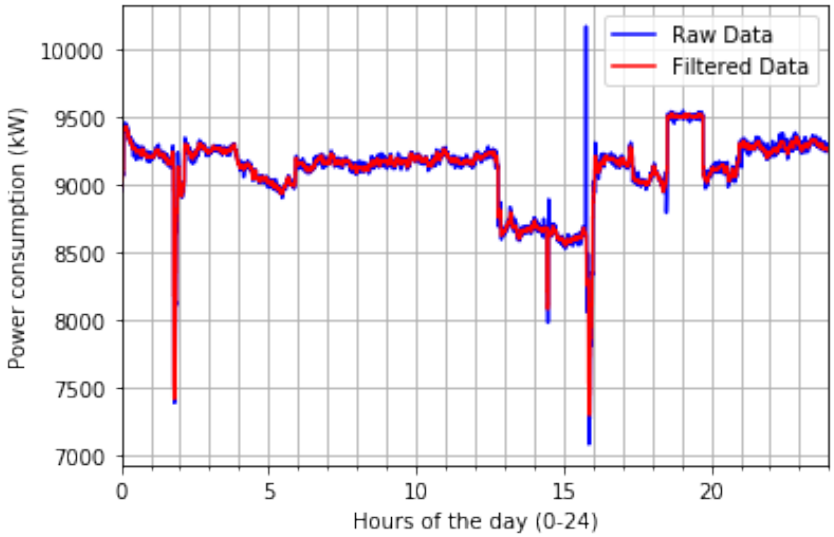

Figure 4.1: One day's active power consumption data for randomly picked vessel/engine and day. Original and median filtered signals are visualized together.

After the median filtering Haar wavelet transformation was applied. Decomposition level of the wavelets was also figured out by trying out different values and suitable candidate was 6. Wavelet filtering was done on a daily level. So we gathered 24 hours of data from one sensor and did a median filtering and wavelet reconstruction for that. One example of a filtered signal together with the original one is shown below.

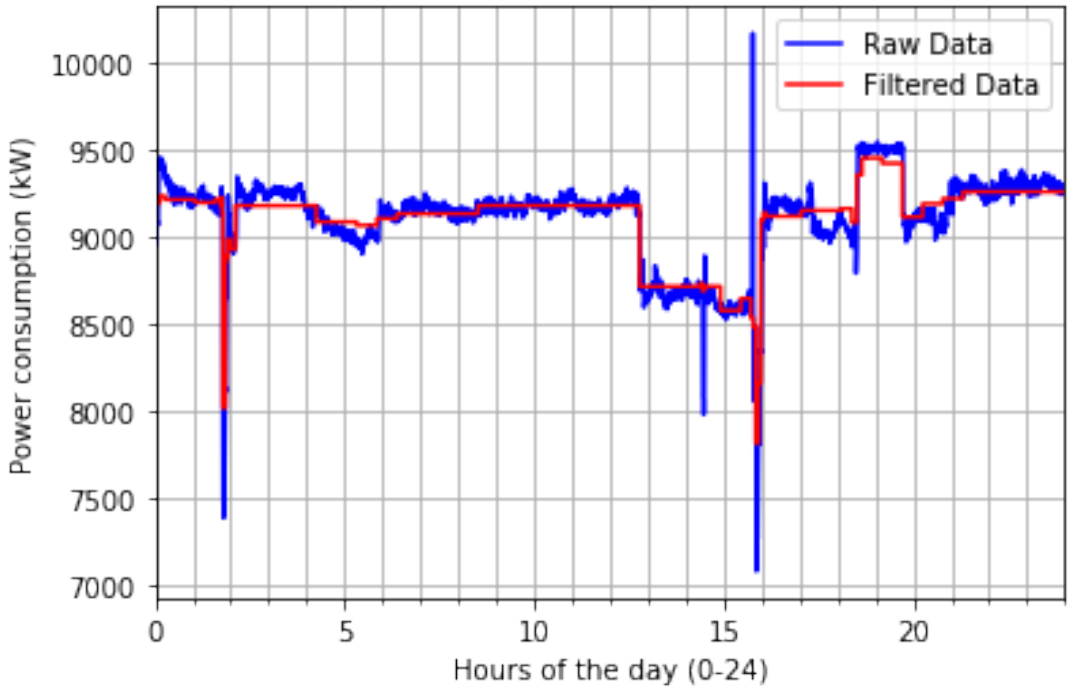

Figure 4.2: Same signal after wavelet filtering

Detail coefficients can be plotted to visualize the building blocks of the reconstructed signal. Below is shown the detail coefficients of each level starting from the lowest graph, which represents the level one coefficient.

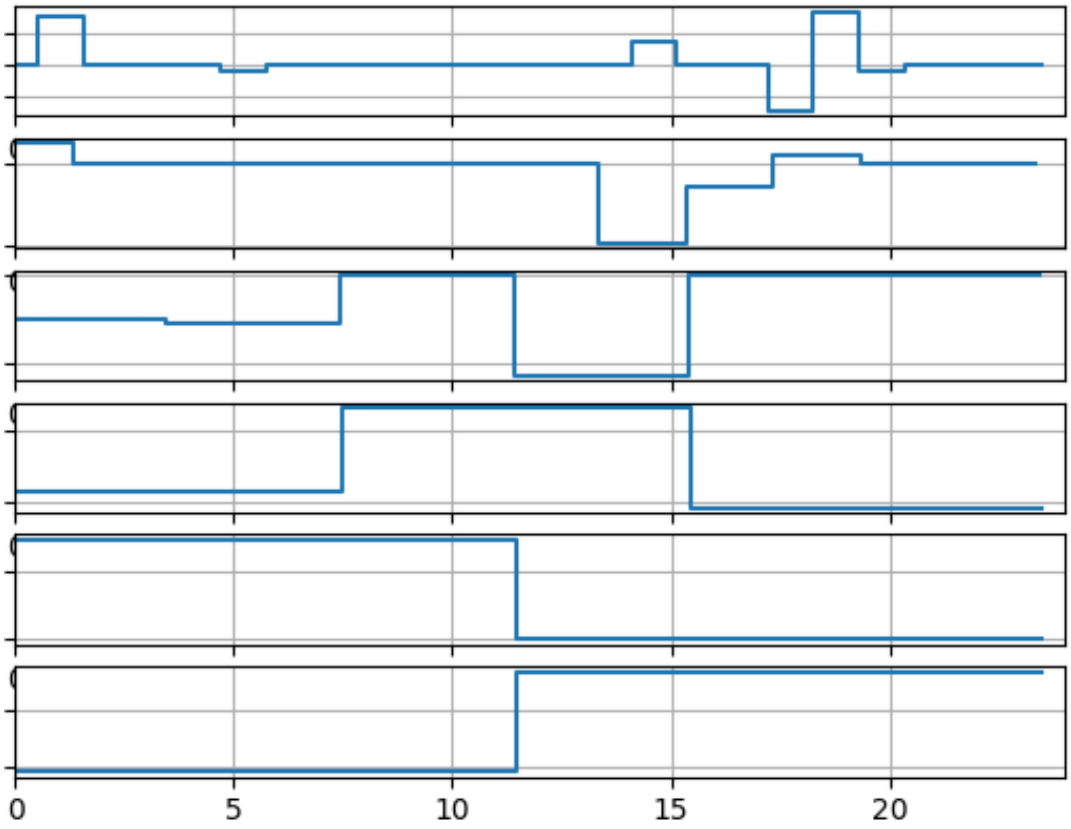

Figure 4.3: Haar wavelet coefficients

In this example the number of data points needed to represent the filtered signal was 43 whereas the length of the vector representing the original signal was 2882. It means almost 99 % drop in the data size. Still when visually inspected, the information we get from the filtered signal is at least as valuable for our case as what the original signal gives us.

4.3 Combining data sets

The power consumption data from sensors is combined with AIS location data to one fact table in a relational database. The data sets are combined based on timestamp and IMO-number with the so called "asof merge", which means that instead of using equal match on timestamps we match on nearest timestamp. We'll do it so that we take each

change in the power value and combine that with the closest following AIS data point. If there's no AIS data within the following hour, the data from that timestamp is discarded.

Clarksons vessel master data is coming from a different source and it is also tabular data. It can be combined to the fact table with IMO-numbers as keys.

4.4 The power consumption estimation

We will construct a model to estimate the power consumption of a vessel and compare the results and accuracy to the results given by admiralty coefficient formula. As mentioned earlier, we will test the model performance with one vessel and use four other vessels to train the model.

4.4.1 Comparison of methods

We started by selecting one vessel randomly out of the five vessels to which we will do the estimations. The data of the remaining four vessels will be used to train the models and sensor data of the fifth vessel will be kept hidden until the model performance evaluation. Mean absolute percentage error is set as the accuracy metric for all of the measurements.

We started by doing the estimations with admiralty coefficient formula to set the benchmark. We don't have information about the actual block coefficients of the vessels so we will use a rough average of typical block coefficients for cruise ships, which is 0,7. [14]

The mean absolute percentage error was calculated to be 50 % for the target vessel. Below is visualized the measured points and relations of predicted and real values. The closer to the red line the point is, the better the prediction.

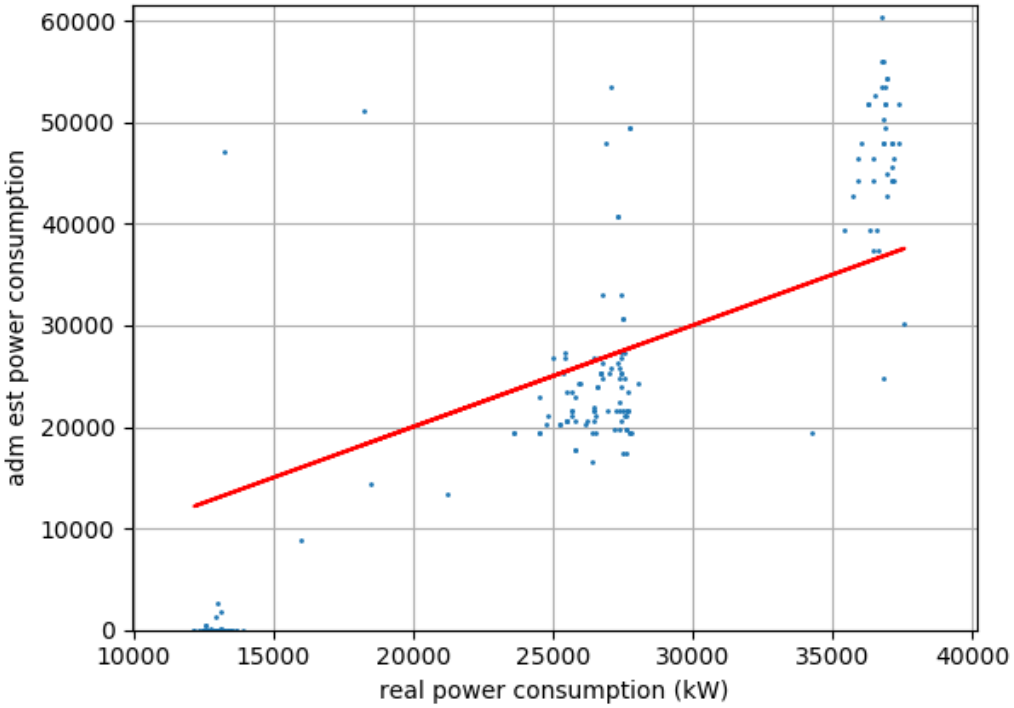

Figure 4.4: Power consumption estimation with admiralty formula

General impression about the admiralty formula estimations is that higher values are estimated too high and lower values are estimated too low. There is also quite a lot of variance especially in the higher values.

Next we tested a simple linear regression model to estimate the powers. We collected training data from four vessels. That included total of 3914 data points. Test data from one remaining vessel consisted of 199 data points. Training data was normalized before optimizing the model. The visualization of model performance is shown below.

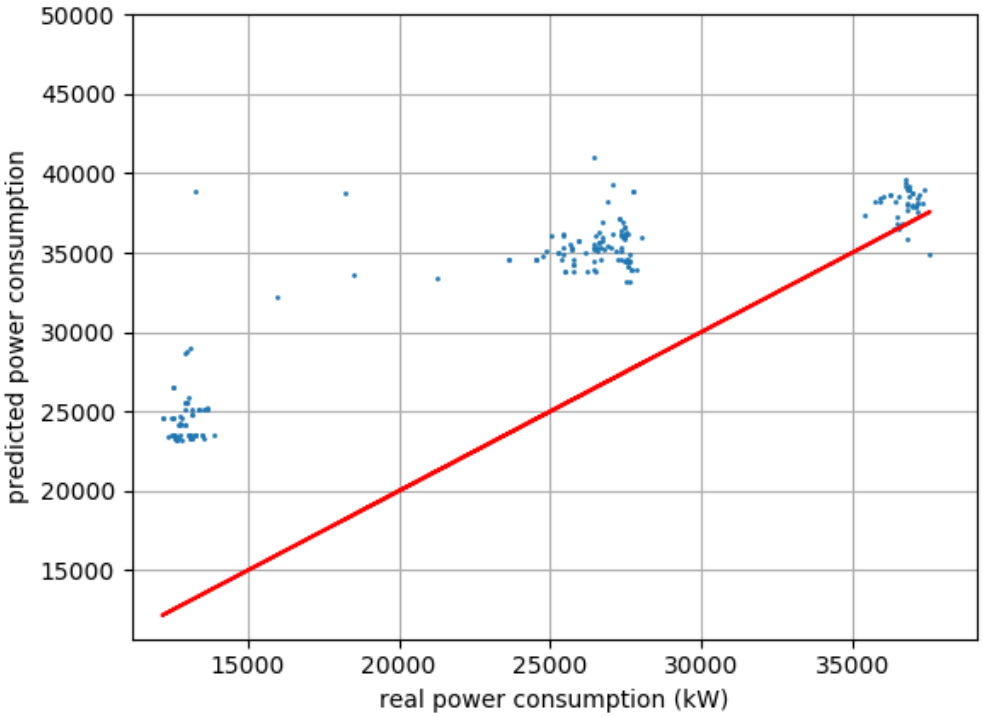

Figure 4.5: Power consumption estimation with linear regression

Now the mean absolute percentage error is 46,5 % so only slightly better than what the empirical equation gave. On the other hand, now the problem is more on the bias than on the variance side. So the points are nicely clustered together but the predictions are systematically wrong in lower power areas.

Next approach was to apply lasso regularization to the linear model. The optimization of regularization coefficient was done by searching the optimal α with exhaustive cross-validation.

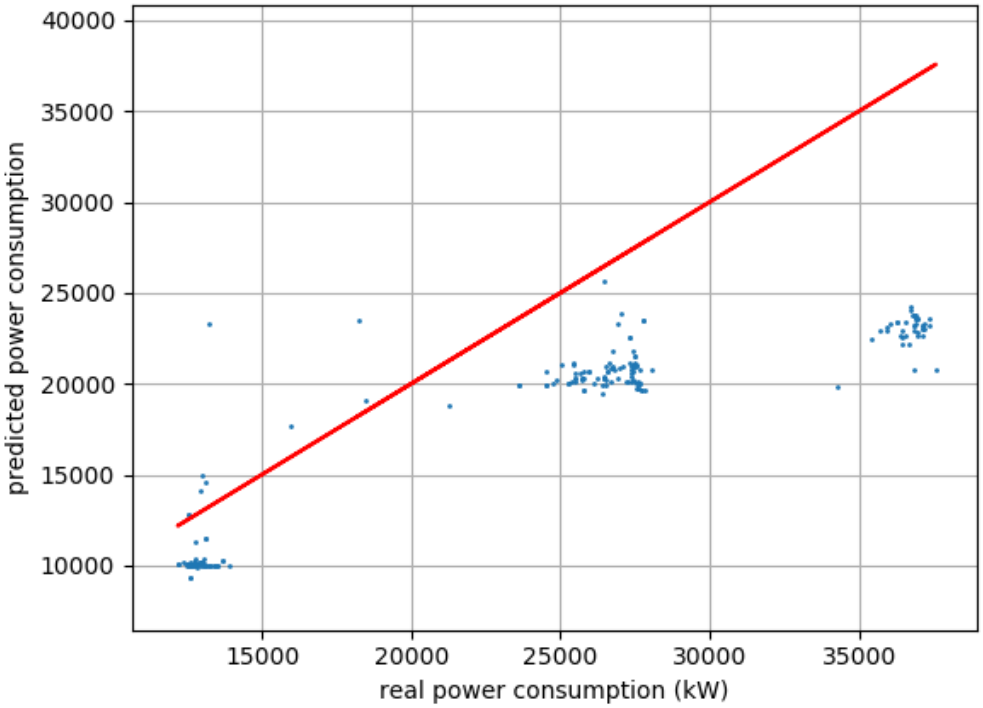

Figure 4.6: Power consumption estimation with lasso regression

Mean absolute percentage error now is 25 %.

When visually inspected, the difference between errors in linear regression and lasso doesn't seem that big. One reason for the confusion might be that MAPE is asymmetric such that equal errors above the actual value result in a greater value than those below the actual value and it puts a heavier penalty on forecasts that exceed the actual than those that are less than the actual. For example, the MAPE is bounded on the low side by an error of 100 %, but there is no bound on the high side.[13]

Nevertheless, it seems that linear models are not suitable for this problem. For example, it is known fact that power consumption is not linearly dependent on speed. On the contrary, the dependence is cubic like seen from the admiralty coefficient.

Polynomial regression was considered as one solution but even second degree polynomial regression proved to be computationally a little bit too challenging to execute in reasonable time with the selected set of features and computing power. This happens because polynomial regression is considered as a special case of linear regression where polynomial features are constructed from the coefficients in a way that the number of fea-

tures becomes quadratic compared to the original number of features, which in this case becomes significant because of the categorical features. The original number of features when categorical variables are one-hot encoded is already 162.

Random forest regression was chosen as the non-linear model because it is relatively easy to understand but really powerful algorithm that works well in most of the cases. As well as in linear algorithms, also here mean square error is used for optimizing the model.

We did an exhaustive search over sets of parameters to find the optimal parameter combination for random forest model. We tried different combinations of following parameters: number of trees in the forest, the number of features to consider when looking for the best split, the minimum number of samples required to split an internal node and whether or not to use bootstrap aggregation.

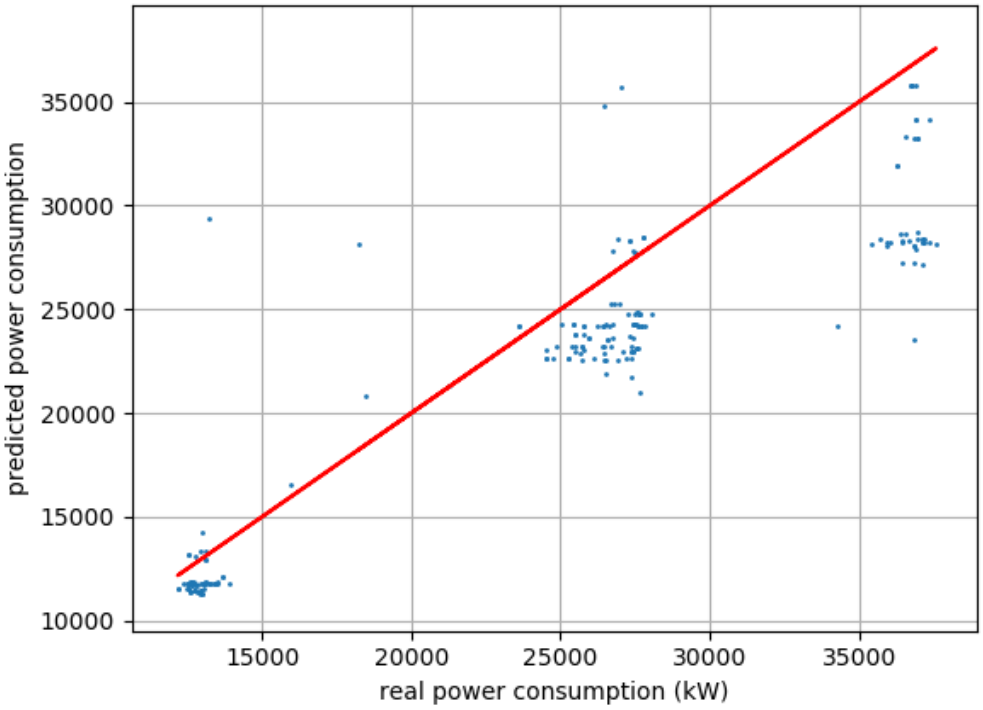

Figure 4.7: Power consumption estimation with random forest regression

Now the mean absolute percentage error is 12,4 % so the result is becoming significantly better than in linear models.

We are able to print out the importance of each feature in determining the splits of

decision trees. Feature importances are shown in the table below:

Variable	**Importance**
sog	77.7 %
length	5.1 %
latitude	3.6 %
longitude	3.0 %
breadth	1.2 %
draught	1.1 %
heading	0.3 %
cog	0.2 %
flag_country	0 %
nav_status	0 %
destination	0 %
callsign	0 %

Table 4.1: Feature importances in random forest estimations

We'll see that there are a lot of varibles that don't bring much value to our model. Reducing the dimensionality seems like a good option in this case. It would reduce overfitting and also decreases the computing time in the optimization process. We created all the models again after dropping out half of the features. So the selected features for the final model are sog, length, latitude, longitude, breadth and draught. Now we were able to create also a second order polynomial regression model because we dropped all the categorical features among other things.

Now the results were

Model	**MAPE**
Linear regression	12,6 %
Lasso	12,9 %
Polynomial regression	11,7 %
Random Forest	10,4 %

Table 4.2: Mean absolute percentage errors of different models

Now the results look good also with linear models. Lasso regularized model is now the worst performing model but still pretty close to others in performance. Lasso in this case works in a way that it simplifies the model by dropping two features completely;

longitude and draught. Intuitively, longitude doesn't tell much about the power demand. Latitude tells more about temperature at least because weather might be warmer closer to equator. Draught on the other hand usually tells a lot about power demand but in this case we have vessels with similar and relatively static draughts so that doesn't work as an explanatory variable with the data at hand.

The model of choise is random forest regressor with bootstrap aggregation. Following parameters were optimized to the model with exhaustive search:

Parameter	**Optimal value**
The number of features to consider when looking for the best split	$\sqrt{n}$, where n is the total number of features
The minimum number of samples required to split an internal node	8
The number of trees in the forest	30

Table 4.3: Optimized hyper parameters for random forest

The feature importances in the final model are

Variable	**Importance**
sog	57.0 %
latitude	20.2 %
length	6.5 %
breadth	6.1 %
longitude	5.3 %
draught	4.9 %

Table 4.4: Feature importances of final model selection

The importances of different features may change a lot when adding more data from varying types of vessels. Assumably this model works well only with large cruise vessels.

We are also able to inspect the individual decision trees in great detail. From the picture below we can see how the iterative process works in decision trees. In this part of the tree the model is optimizing the prediction based on vessel length and latitude value. This is just a small fraction of the whole tree. Feature values are standardized, which makes it a bit difficult to interpret the actual effect of the parameters. Left hand arrows

indicate 'TRUE' and right hand arrows indicate 'FALSE'. One interesting finding from this image alone might be that power consumption would be lower if latitude is below certain limit, which means that the vessel is travelling in south. Of course this is just one node of the tree and should not lead to any big conclusions.

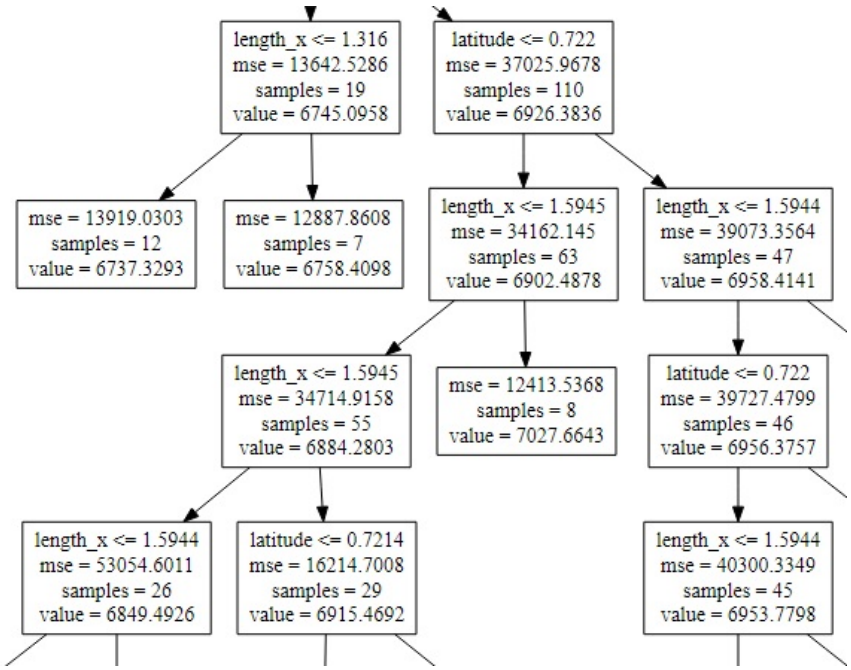

Figure 4.8: Small part of one of the actual decision trees in the forest

The other point of interest is the estimation of fuel consumption. It goes pretty much hand in hand with power demand estimation so the same feature selections should work but fuel consumption is probably a bit more intuitive performance indicator and it makes it easier to calculate for example the operational costs of the vessel. So we will create the random forest regression model again with fuel consumption as the target variable.

Mean absolute percentage error of the fuel consumption prediction is 17,6 % and the feature importances

Variable	Importance
sog	63.3 %
latitude	15.6 %
breadth	6.3 %
length	6.1 %
longitude	5.9 %
draught	2.9 %

Table 4.5: Feature importances of random forest model for fuel consumption estimation

As expected, the features effect in a pretty similar way to fuel consumption as to power consumption.

4.4.2 Power profile

Considering the earlier presented scenario, where we want to estimate the power demand of for example some new build vessel when we know the ship measures and we can assume it's operating profile based on some sister vessels, we can predict the power profile of a vessel. In this scenario we can only use speed and draught along with static vessel data as features. We will optimize a new random forest regressor model with only mentioned features and apply it to the bulker ship we dealt with in the operating profile section.

We can add the power estimation to every element in the speed-draught matrix. For simplicity we will take only two draught areas (ballast and laden) and make separate power profiles for them. First we will see what is the power profile when the ship is not loaded.

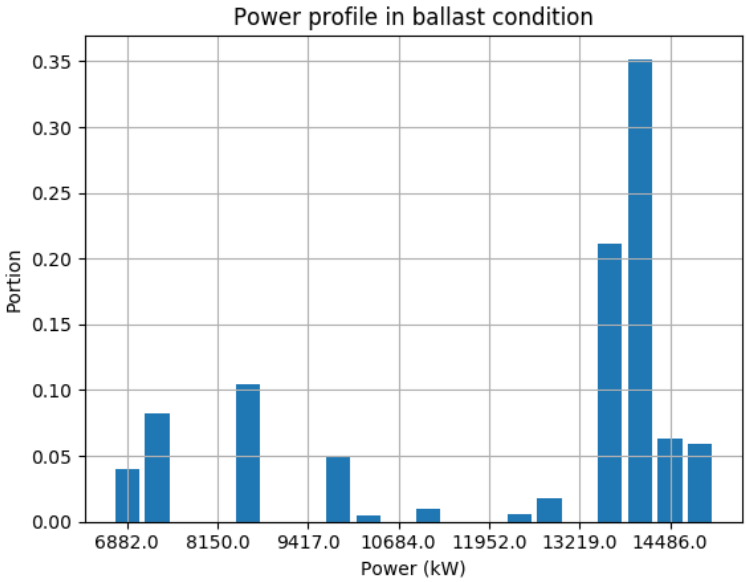

Figure 4.9: Estimated power profile of example vessel in ballast based on speeds of October 2018

Let's visualize also the speed profile since speed is the only non-constant predictor and so it explains the changes in power output.

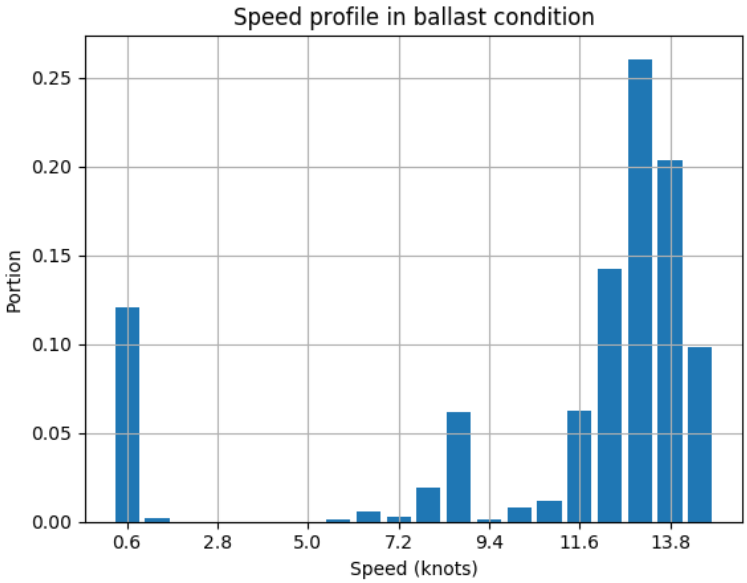

Figure 4.10: Speed profile in ballast condition in October 2018

Same figures when the vessel is loaded look like this:

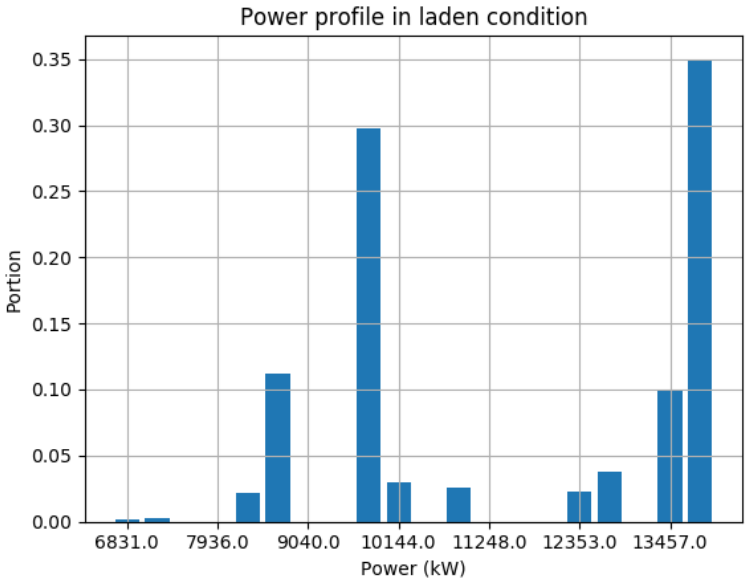

Figure 4.11: Estimated power profile of example vessel in laden based on speeds of October 2018

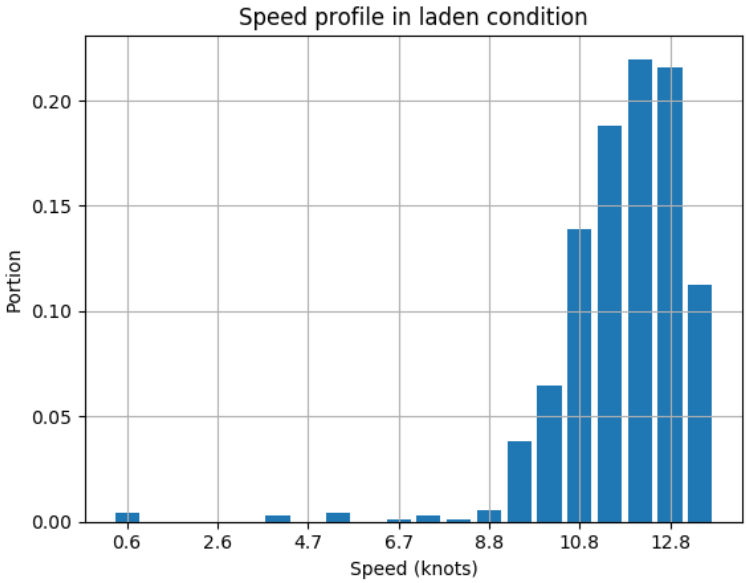

Figure 4.12: Speed profile in laden condition in October 2018

It seems that power consumptions are smaller in laden condition which sounds unintuitive. There are two things that explain this. Like seen in speed profile graphs, speeds are lower when the vessel is loaded. Another thing is that the impact of draught is underestimated by the model because the model was trained with cruise ship data and the draughts of cruise ships don't typically fluctuate.

To make sure that our model reflects to the changes of speed properly, we can visualize the estimated power as function of speed when other features remain constant.

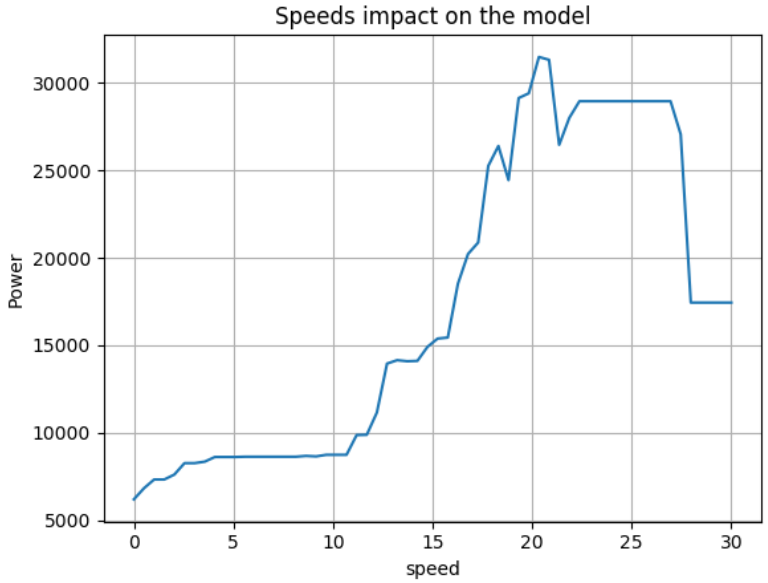

Figure 4.13: Speed vs estimated power of single vessel when other features are constant (draught=14m, length=225m, width=32m)

There seems to be that kind of exponential dependency between speed and power like there should be but after 20 knots the model breaks. That is probably because there's not much examples of speeds above 20 knots in the training data.

Chapter 5

Conclusions

In this study we investigated the possibilities to estimate power demands and fuel consumptions of vessels with machine learning methods. We examined how those methods compare to empirical ways of calculating power demands. In addition we suggested an effective way of compressing some signals coming from the sensors from the vessels' engines.

The main motivation behind this experiment was to be able to estimate the needed propulsion power distribution and fuel consumption for different operating modes for a new build vessel based on planned routes, speeds and other factors. One of the other use cases would be for example finding the optimal route and speeds for the voyage.

Now we had sensor data only from cruise vessels. For the future, we would need data from different vessel types and probably the best way would be to create own models for each vessel type. At least some vessels that have varying draughts, like bulk carriers and cargo vessels should be included.

Bibliography

[1] International Maritime Organization, Third IMO Greenhouse Gas Study 2014 (2015), Retrieved from http://www.imo.org/en/OurWork/Environment/PollutionPrevention/AirPollution/Documents/Third%20Greenhouse%20Gas%20Study/GHG3%20Executive%20Summary%20and%20Report.pdf

[2] Pham, Charlotte Minh-Hà L., Basic Terminology of Shipbuilding, 2012. Retrieved from http://www.unesco.org/new/fileadmin/MULTIMEDIA/HQ/CLT/images/630X300/APPENDIXB.pdf

[3] Kovanen, Lauri, Hull Fouling, Study of full houling on cruise vessels across various seas, 2012. Retrieved from https://www.eniram.fi/wp-content/uploads/2013/11/Hull-fouling-study-1.12MB.pdf

[4] de Mendoza y Ríos, Joseph. Memoria sobre algunos métodos nuevos de calcular la longitud por las distancias lunares: y aplication de su teórica á la solucion de otros problemas de navegacion, 1795

[5] Clarksons Research Portal. Retrieved from https://www.clarksons.net

[6] Bassam, Ameen, Phillips, Alexander, Turnock, Stephen, Wilson, Philip., Ship Voyage Energy Efficiency Assessment Using Ship Simulators. MARINE 2015 - Computational Methods in Marine Engineering VI, 2015.

[7] Babicz, Jan. Wärtsilä Encyclopedia Of Ship Technology, Second edition, 2015

[8] Harris, David and Harris, Sarah., Digital design and computer architecture (2nd ed.), 2013

[9] Turpin, Edward A.; William A. McEwen. Merchant Marine Officers' Handbook (4th ed.). Centreville, Maryland: Cornell Maritime Press, 1980

[10] Haar, Alfréd, Zur Theorie der orthogonalen Funktionensysteme, Mathematische Annalen, 1910

[11] Hastie, Tibshirani, Friedman, The Elements of statistical learning (2nd ed), 2009

[12] Donoho, Johnstone, Ideal Spatial Adaptation by Wavelet Shrinkage, Department of Statistics, Stanford University, Stanford, CA, 94305-4065, U.S.A, 1992, Retrieved from http://statweb.stanford.edu/ imj/WEBLIST/1994/isaws.pdf

[13] Armstrong, Collopy, Makridakis, Another Error Measure for Selection of the Best Forecasting Method: The Unbiased Absolute Percentage Error, 1992

[14] Papanikolaou, Ship Design, 2014

[15] Black, Paul, Greedy Algorithm, Dictionary of Algorithms and Data Structures. U.S. National Institute of Standards and Technology (NIST), 2005